S10 C48 - Thinking ahead

Section 10 - Computational thinking

Monday, October 16, 2023

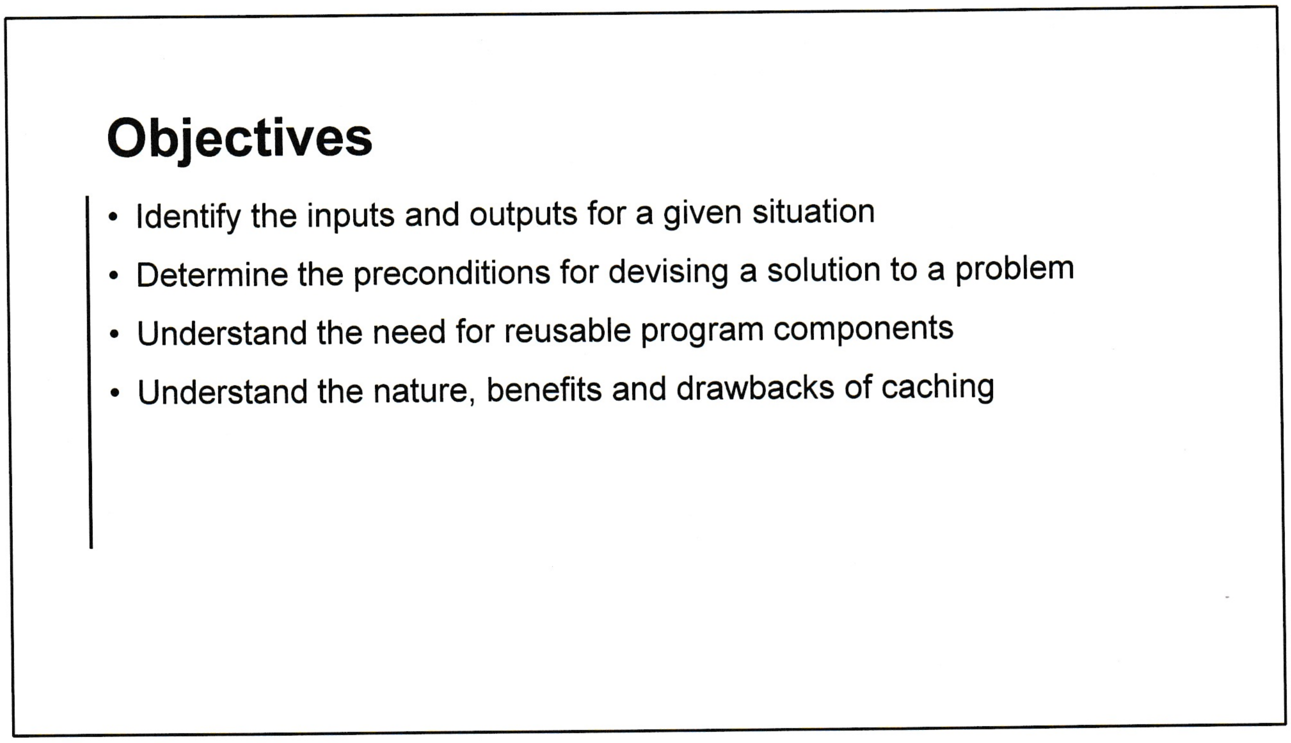
Key terms

Pseudocode

- A notation resembling a simplified programming language, used in program design.

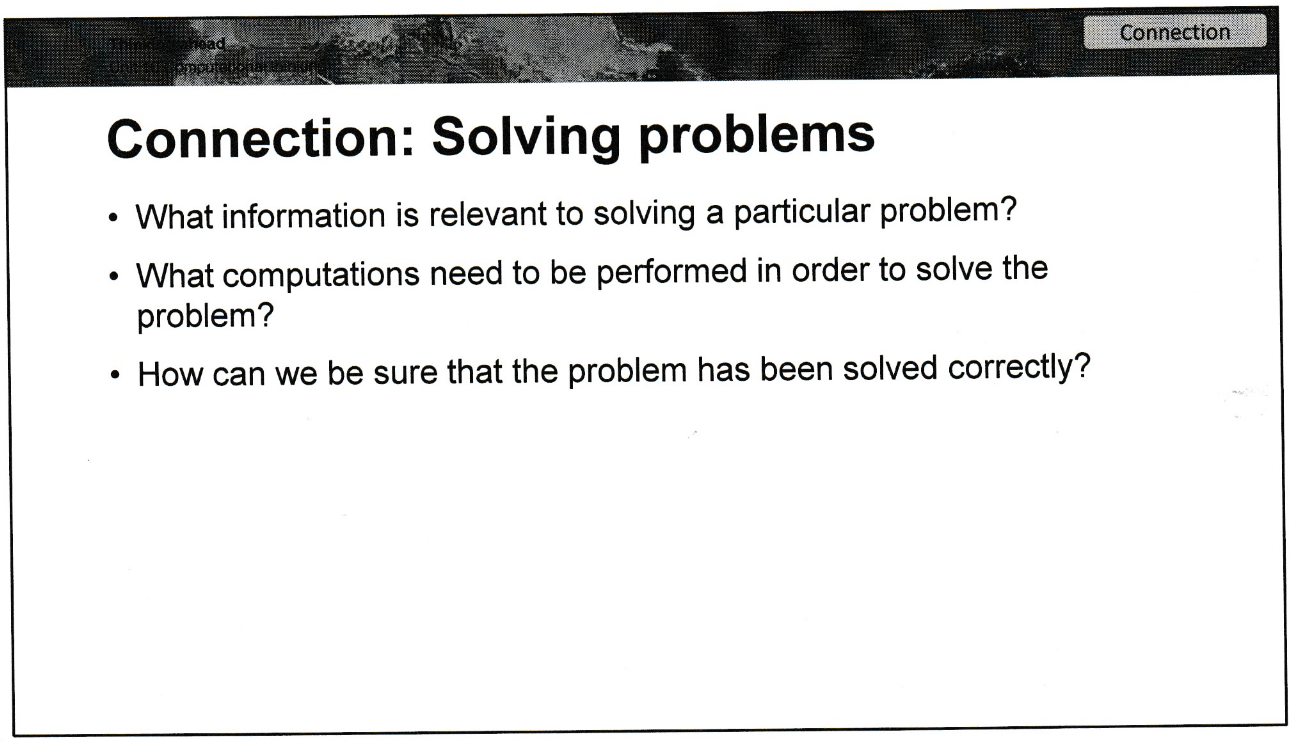
Cache

- The temporary storage of data and instructions



Objectives

- Identify the inputs and outputs for a given situation
- Determine the preconditions for devising a solution to a problem
- Understand the need for reusable program components
- Understand the nature, benefits and drawbacks of caching



Thinking ahead
Unit 10 Computational thinking

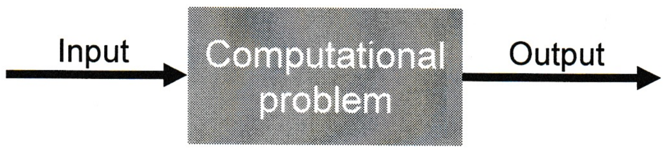
Connection

Connection: Solving problems

- What information is relevant to solving a particular problem?
- What computations need to be performed in order to solve the problem?
- How can we be sure that the problem has been solved correctly?

Devising an abstract model

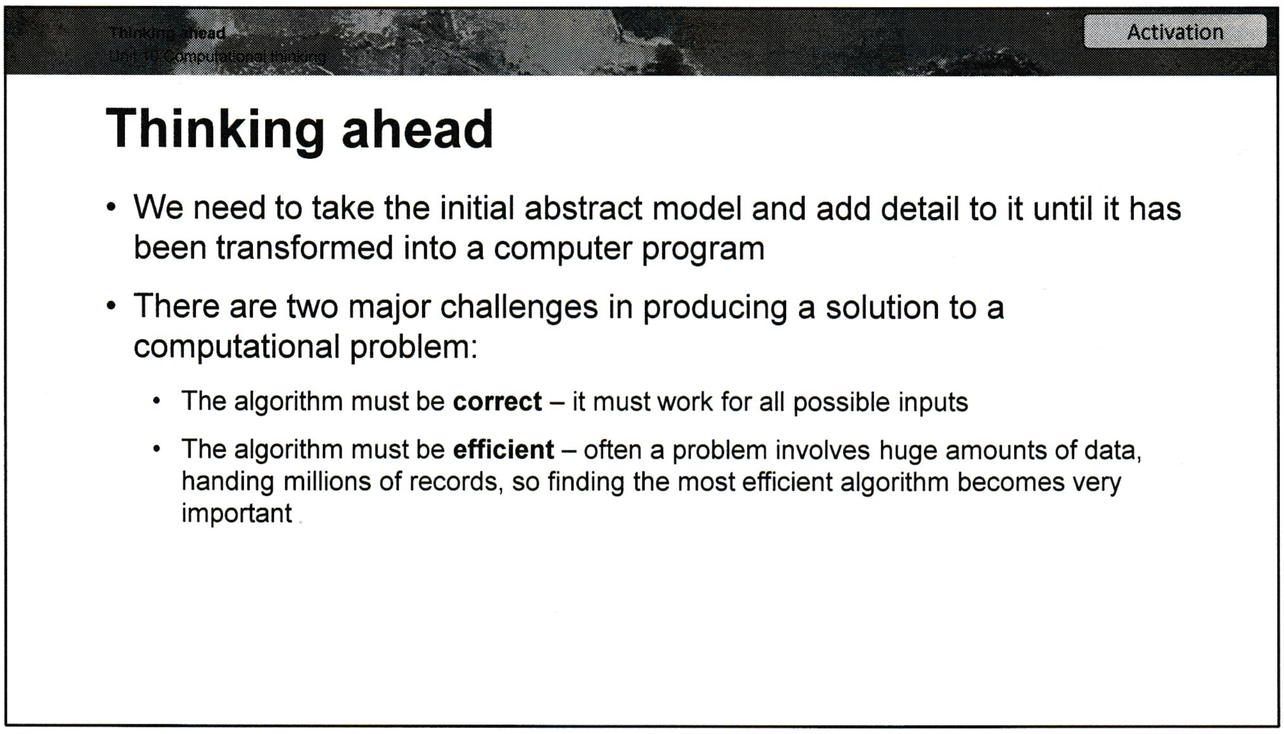
- At its most abstract level, a computational problem can be represented by a simple diagram:



- input is the information relevant to the problem, which could for example be passed as parameters to a subroutine
- output is the solution to the problem, which could be passed back from a subroutine

Q1: Write a pseudocode algorithm which initialises a list of string items, asks the user to enter an item to search for, calls the above function `SearchList` and prints an appropriate message depending on whether the function returns True or False.

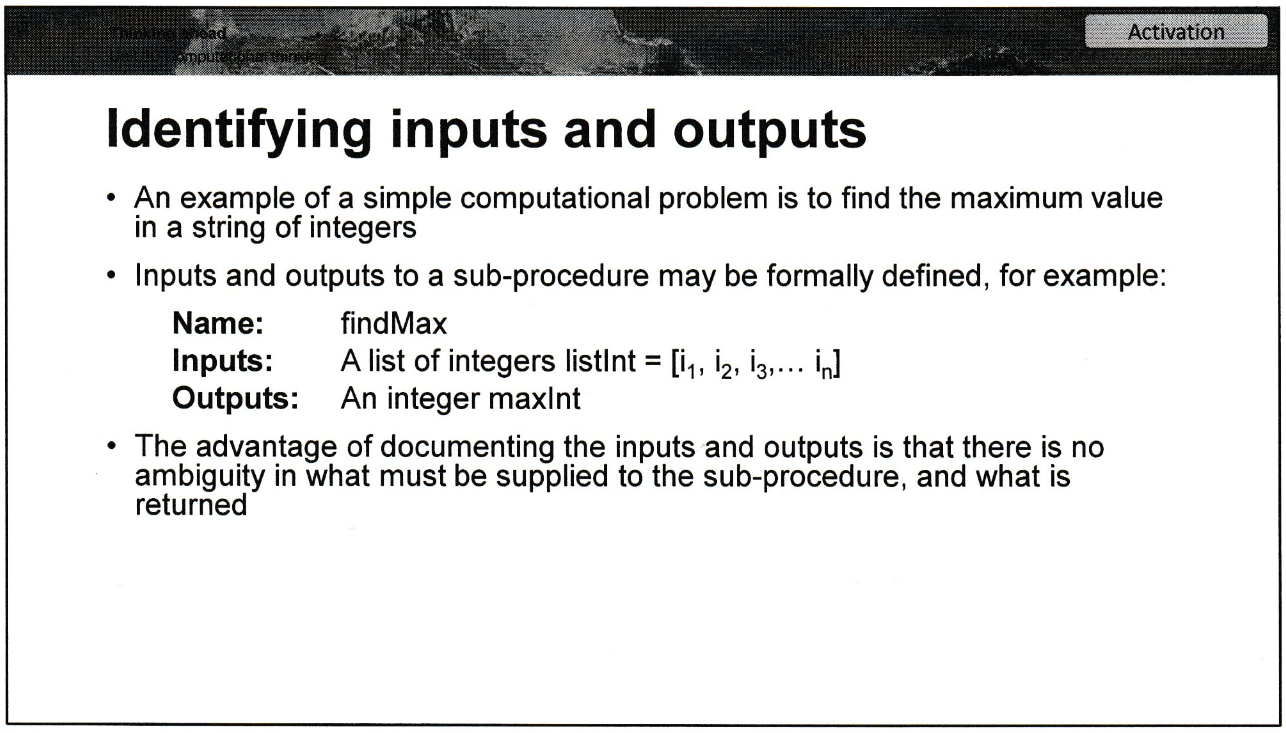
```
List = ['1', '2', '3']
Search = input ('search an item: ')
func searchlist
if res == T :
    print 'found at ' location
else :
    print 'does not exist'
```



Activation

Thinking ahead

- We need to take the initial abstract model and add detail to it until it has been transformed into a computer program
- There are two major challenges in producing a solution to a computational problem:
  - The algorithm must be **correct** – it must work for all possible inputs
  - The algorithm must be **efficient** – often a problem involves huge amounts of data, handing millions of records, so finding the most efficient algorithm becomes very important



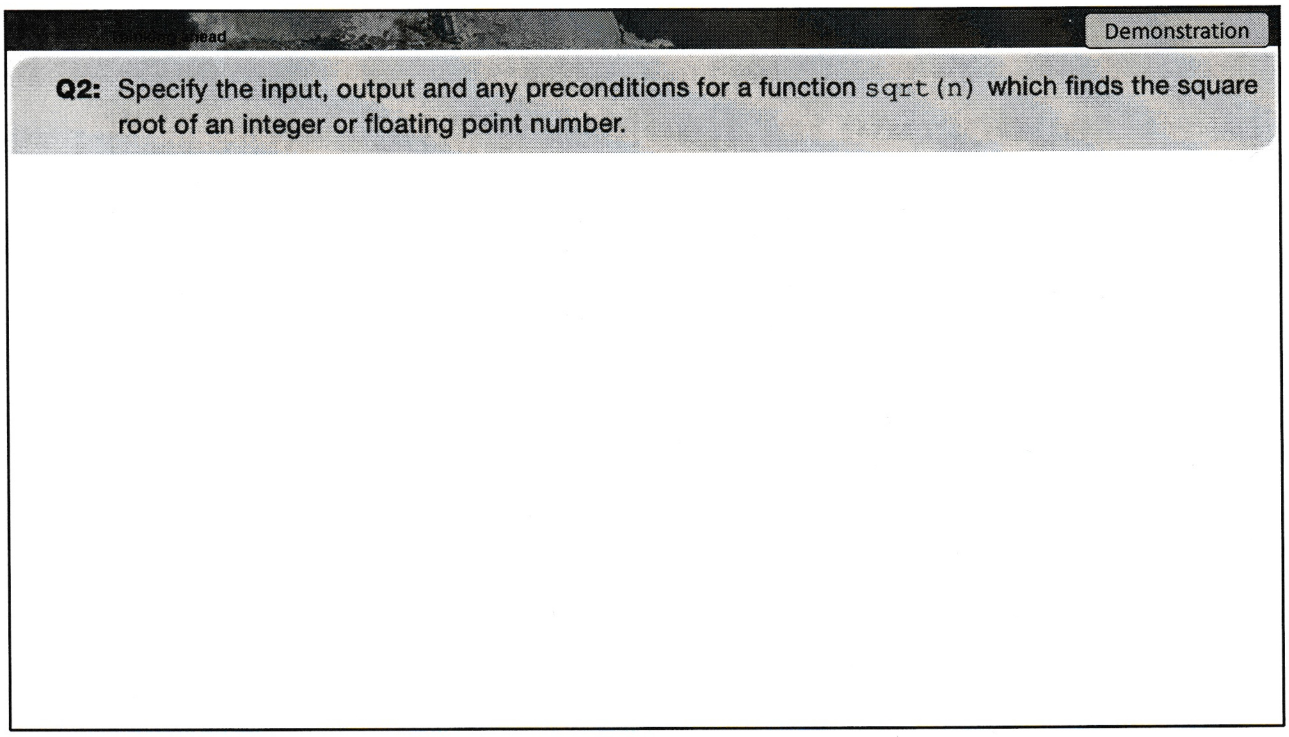
Activation

Identifying inputs and outputs

- An example of a simple computational problem is to find the maximum value in a string of integers
- Inputs and outputs to a sub-procedure may be formally defined, for example:

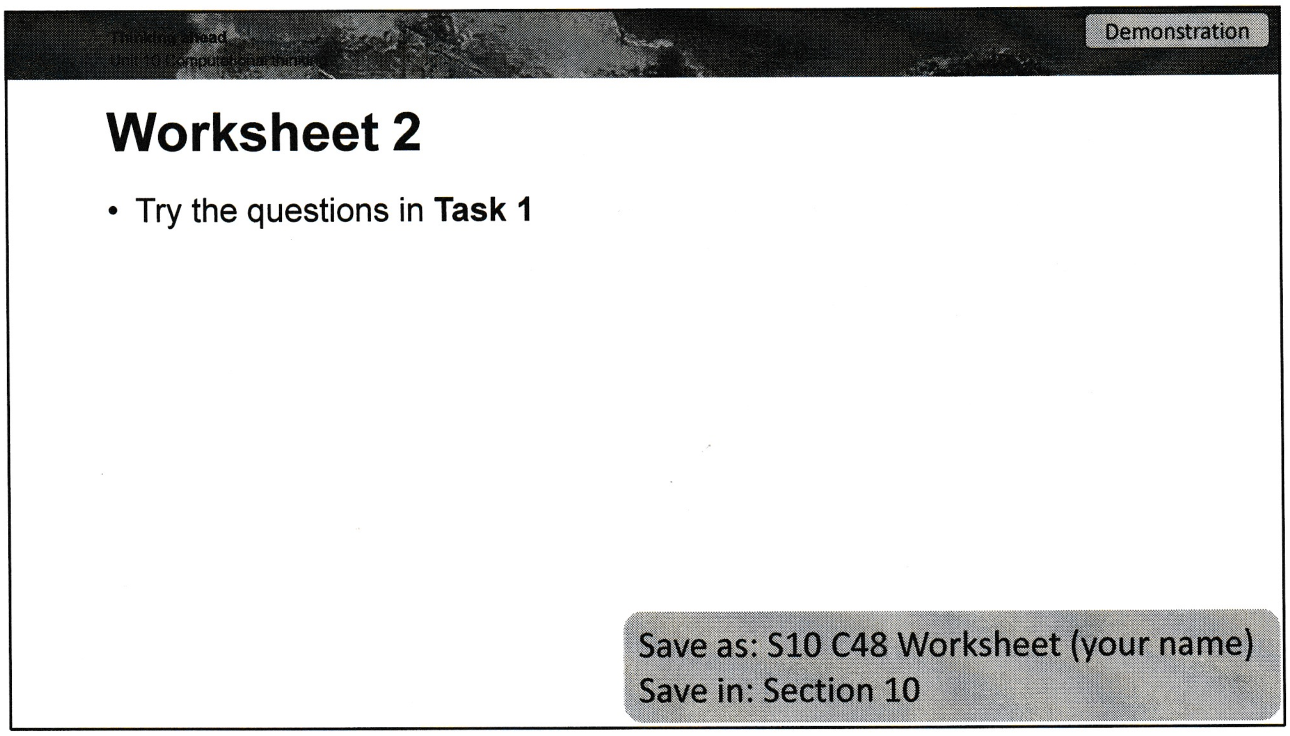
Name:	findMax
Inputs:	A list of integers listInt = $[i_1, i_2, i_3, \ldots i_n]$
Outputs:	An integer maxInt

- The advantage of documenting the inputs and outputs is that there is no ambiguity in what must be supplied to the sub-procedure, and what is returned



Demonstration

Q2: Specify the input, output and any preconditions for a function `sqrt(n)` which finds the square root of an integer or floating point number.

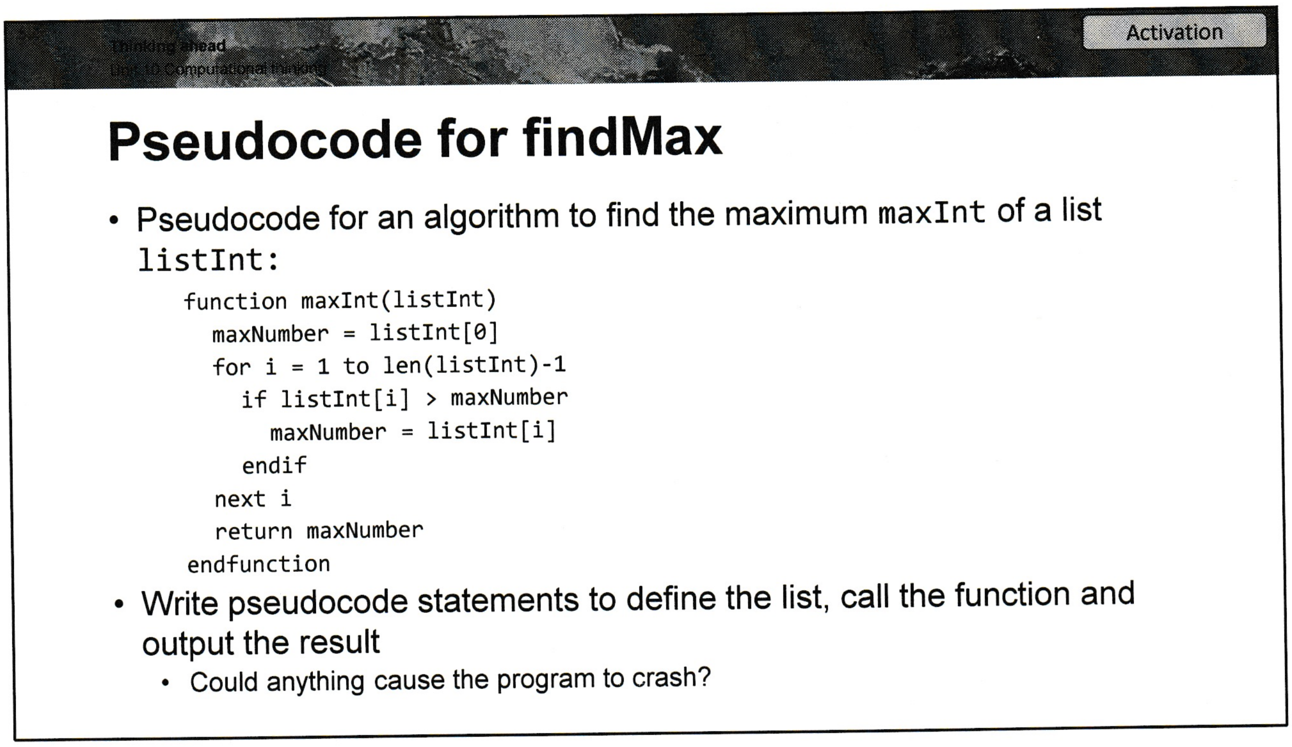


Demonstration

Worksheet 2

- Try the questions in **Task 1**

Save as: S10 C48 Worksheet (your name)
Save in: Section 10



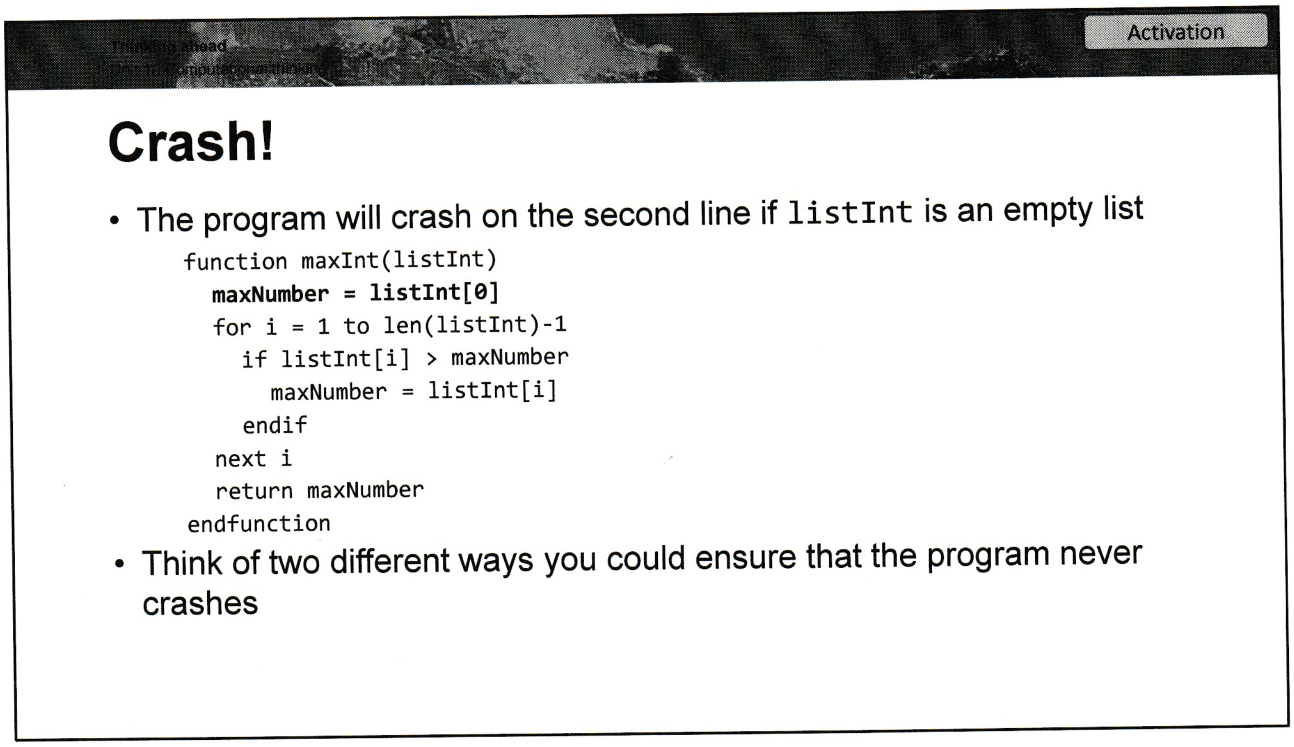
Activation

Pseudocode for findMax

- Pseudocode for an algorithm to find the maximum `maxInt` of a list `listInt:`

```
function maxInt(listInt)
  maxNumber = listInt[0]
  for i = 1 to len(listInt)-1
    if listInt[i] > maxNumber
      maxNumber = listInt[i]
    endif
  next i
  return maxNumber
endfunction
```

- Write pseudocode statements to define the list, call the function and output the result
  - Could anything cause the program to crash?



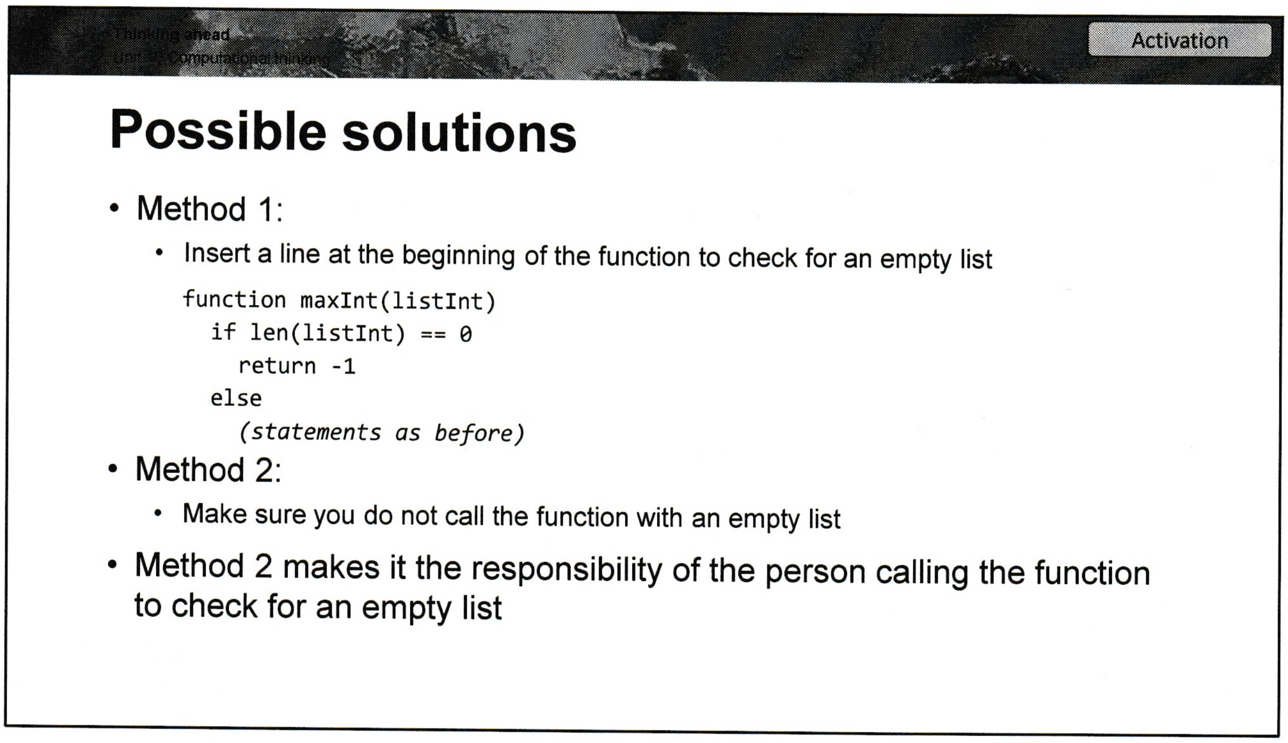
Activation

Crash!

- The program will crash on the second line if `listInt` is an empty list

```
function maxInt(listInt)
  maxNumber = listInt[0]
  for i = 1 to len(listInt)-1
    if listInt[i] > maxNumber
      maxNumber = listInt[i]
    endif
  next i
  return maxNumber
endfunction
```

- Think of two different ways you could ensure that the program never crashes



Thinking ahead
Unit 10 Computational thinking

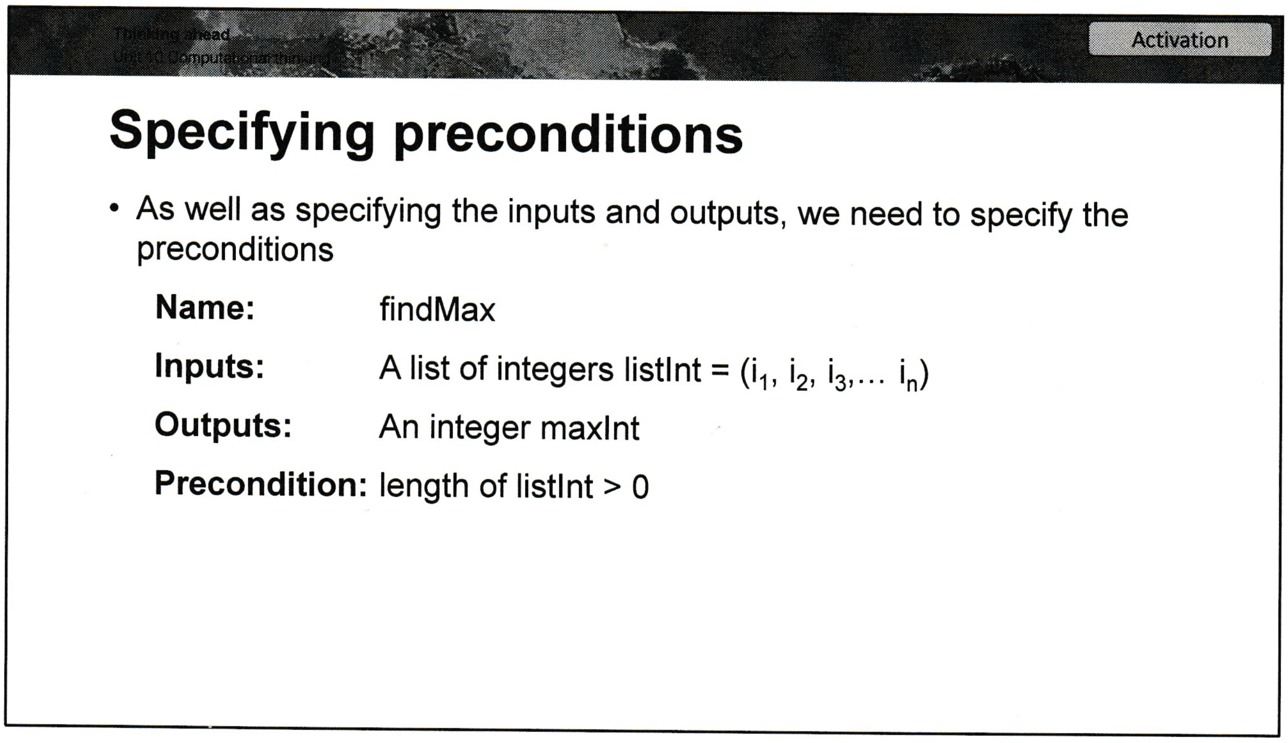
Activation

Possible solutions

- Method 1:
  - Insert a line at the beginning of the function to check for an empty list

```
function maxInt(listInt)
  if len(listInt) == 0
    return -1
  else
    (statements as before)
```

- Method 2:
  - Make sure you do not call the function with an empty list
- Method 2 makes it the responsibility of the person calling the function to check for an empty list



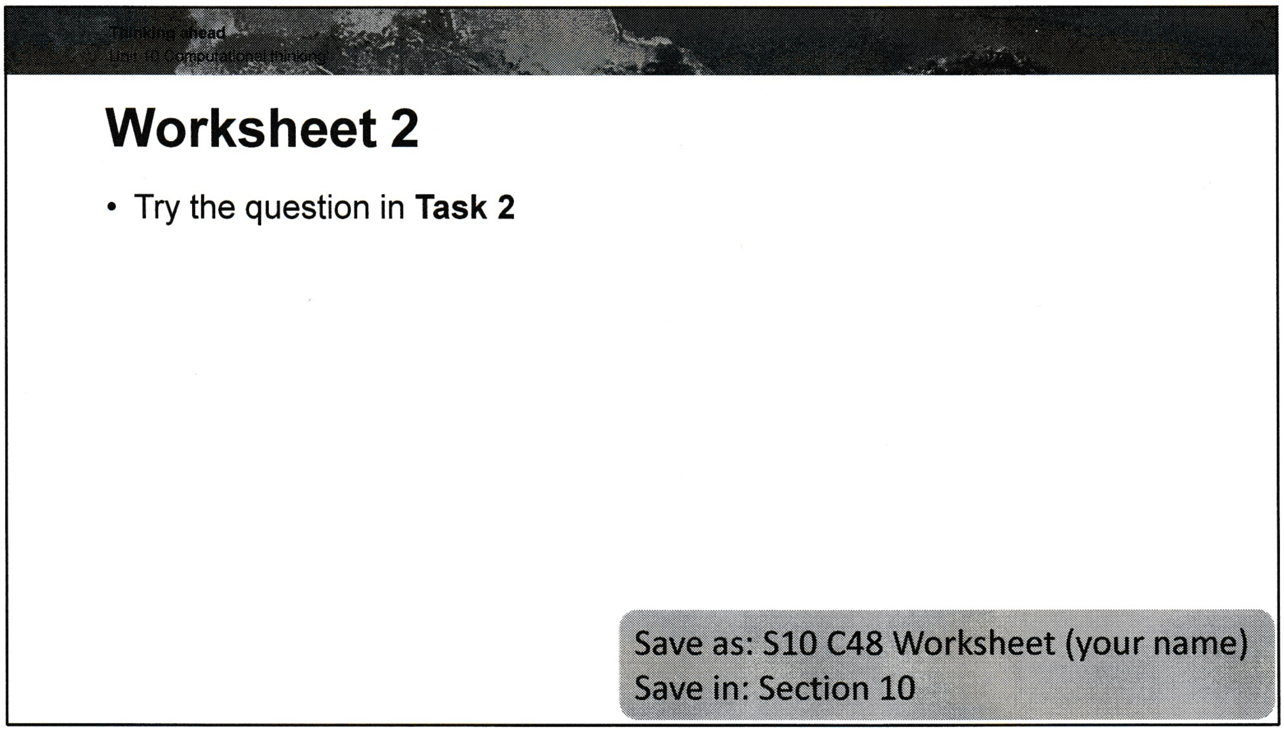
Thinking ahead
Unit 10 Computational thinking

Activation

Specifying preconditions

- As well as specifying the inputs and outputs, we need to specify the preconditions

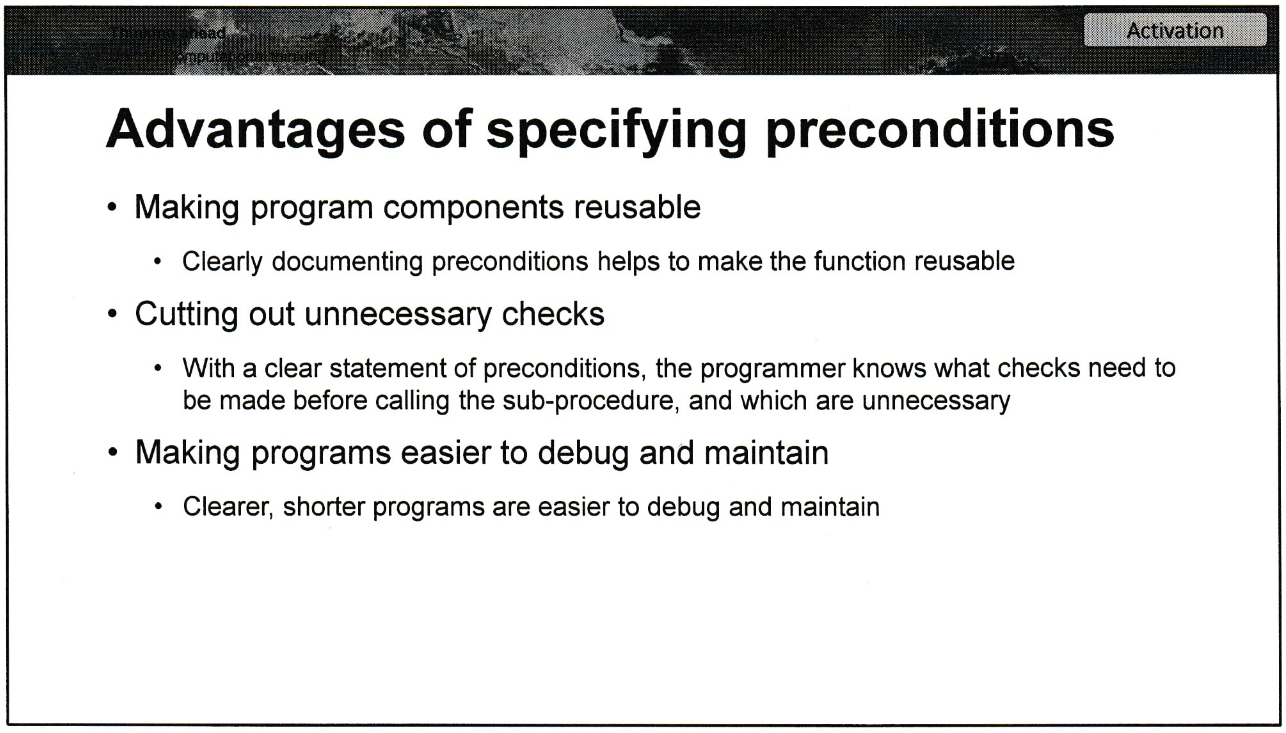
Name:	findMax
Inputs:	A list of integers listInt = (i_1, i_2, i_3,… i_n)
Outputs:	An integer maxInt
Precondition:	length of listInt > 0



Worksheet 2

- Try the question in **Task 2**

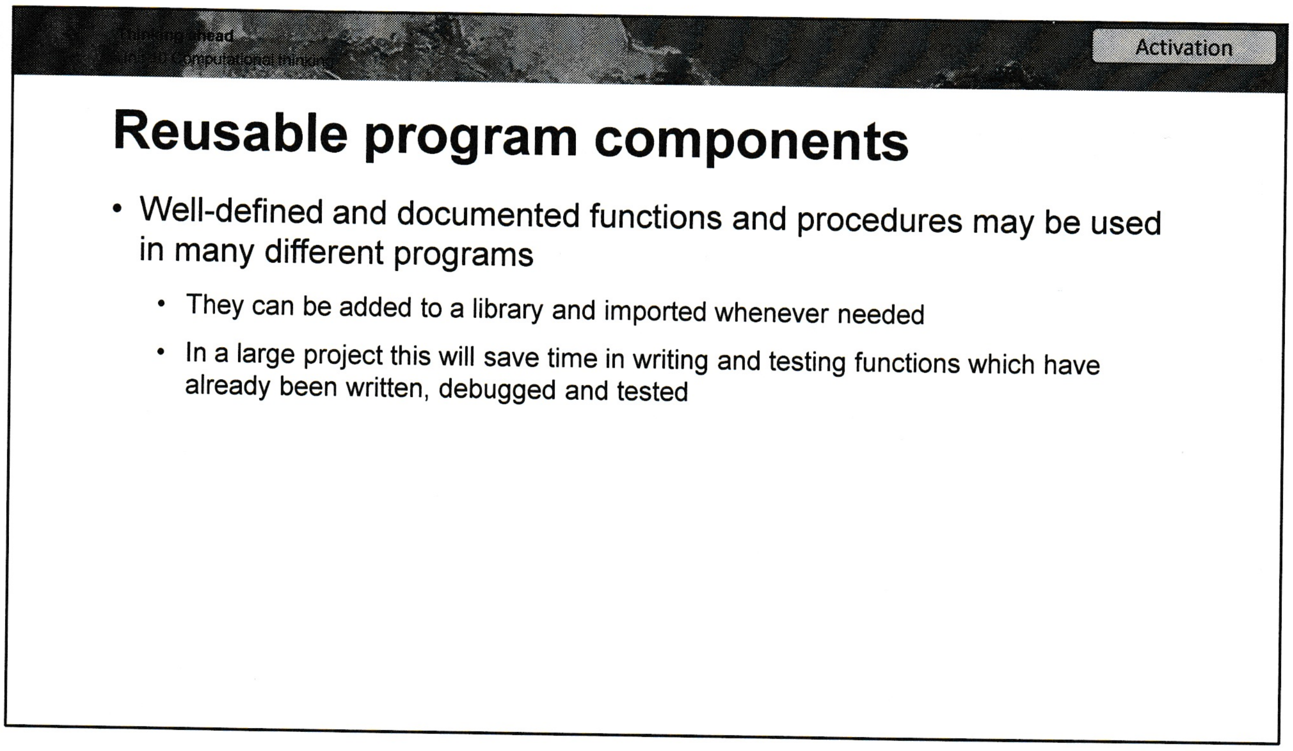
Save as: S10 C48 Worksheet (your name)
Save in: Section 10



Activation

Advantages of specifying preconditions

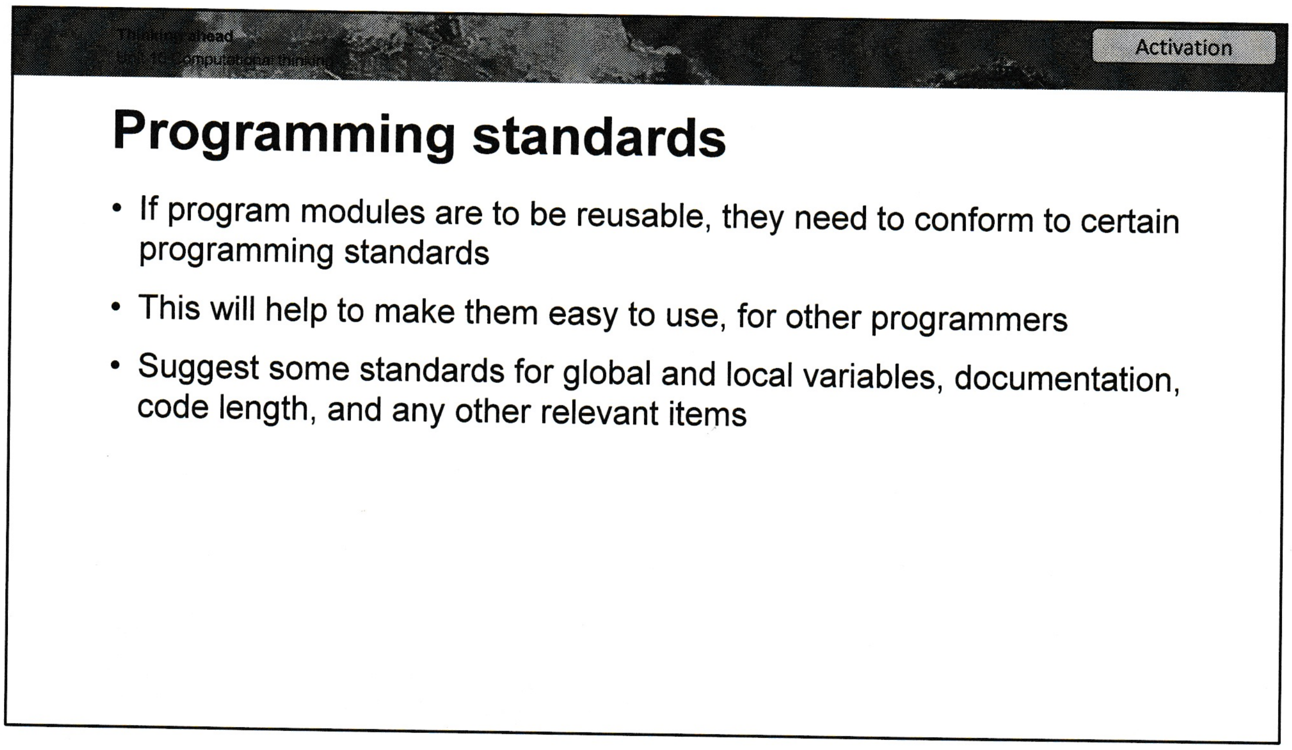
- Making program components reusable
  - Clearly documenting preconditions helps to make the function reusable
- Cutting out unnecessary checks
  - With a clear statement of preconditions, the programmer knows what checks need to be made before calling the sub-procedure, and which are unnecessary
- Making programs easier to debug and maintain
  - Clearer, shorter programs are easier to debug and maintain



Activation

Reusable program components

- Well-defined and documented functions and procedures may be used in many different programs
  - They can be added to a library and imported whenever needed
  - In a large project this will save time in writing and testing functions which have already been written, debugged and tested



Activation

Programming standards

- If program modules are to be reusable, they need to conform to certain programming standards
- This will help to make them easy to use, for other programmers
- Suggest some standards for global and local variables, documentation, code length, and any other relevant items

Activation

Programming standards

- Here are some typical standards for reusable modules
  - Inputs, outputs and preconditions should be documented
  - Variable identifiers should conform to a standard convention
  - All variables must be local to the module
  - Documentation should be in a standard format, explaining what the module does, who it was written by, date it was written
  - Explanations of certain sections of code should be included where necessary
  - No module should exceed one page of code

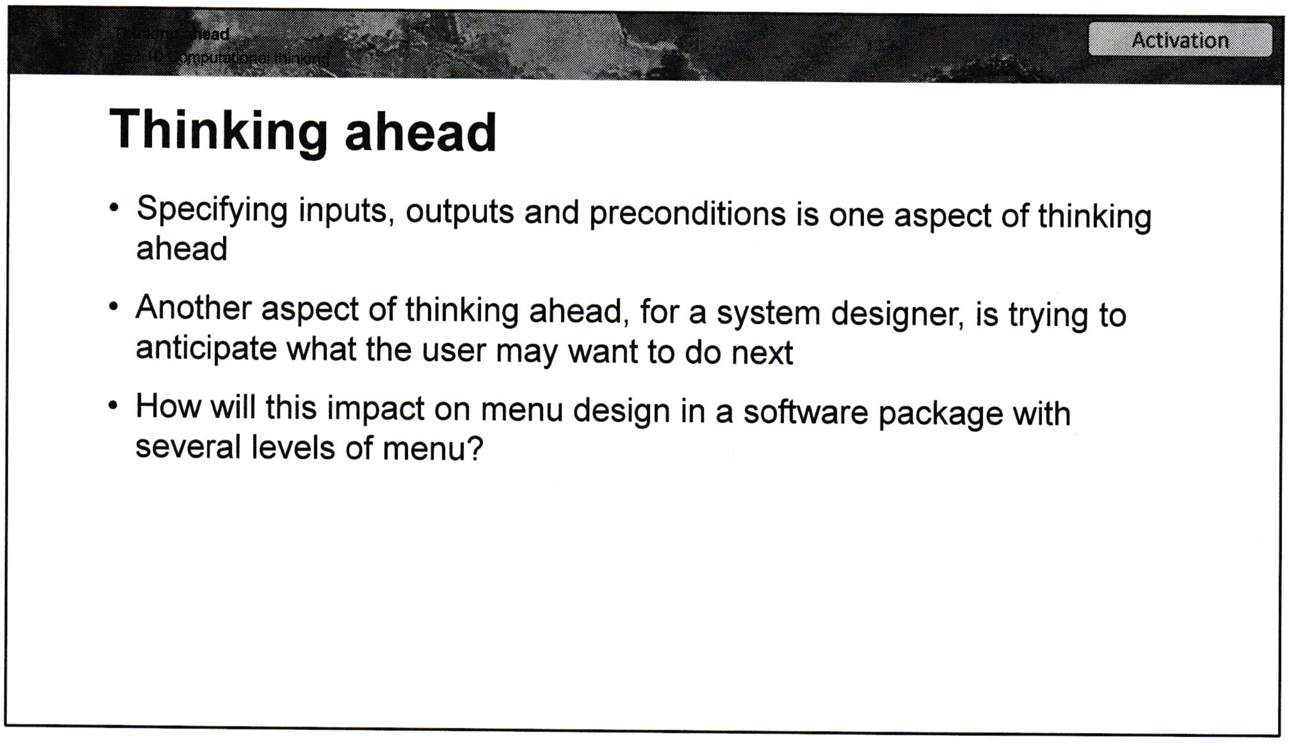
Worksheet 2

- Complete **Task 2**

$$jList = [j_1, j_2, j_3, \dots j_n]$$

$$kList = [k_1, k_2, k_3, \dots k_n]$$

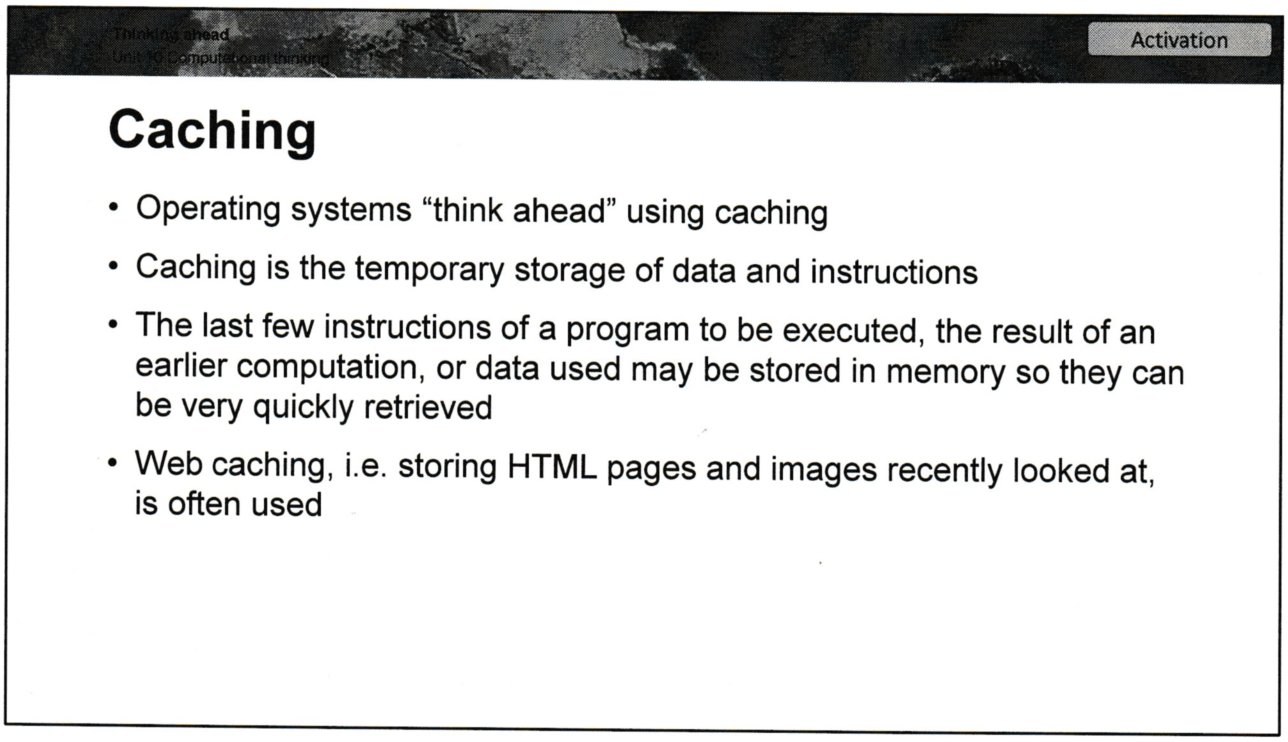
Save as: S10 C48 Worksheet (your name)
Save in: Section 10



Activation

Thinking ahead

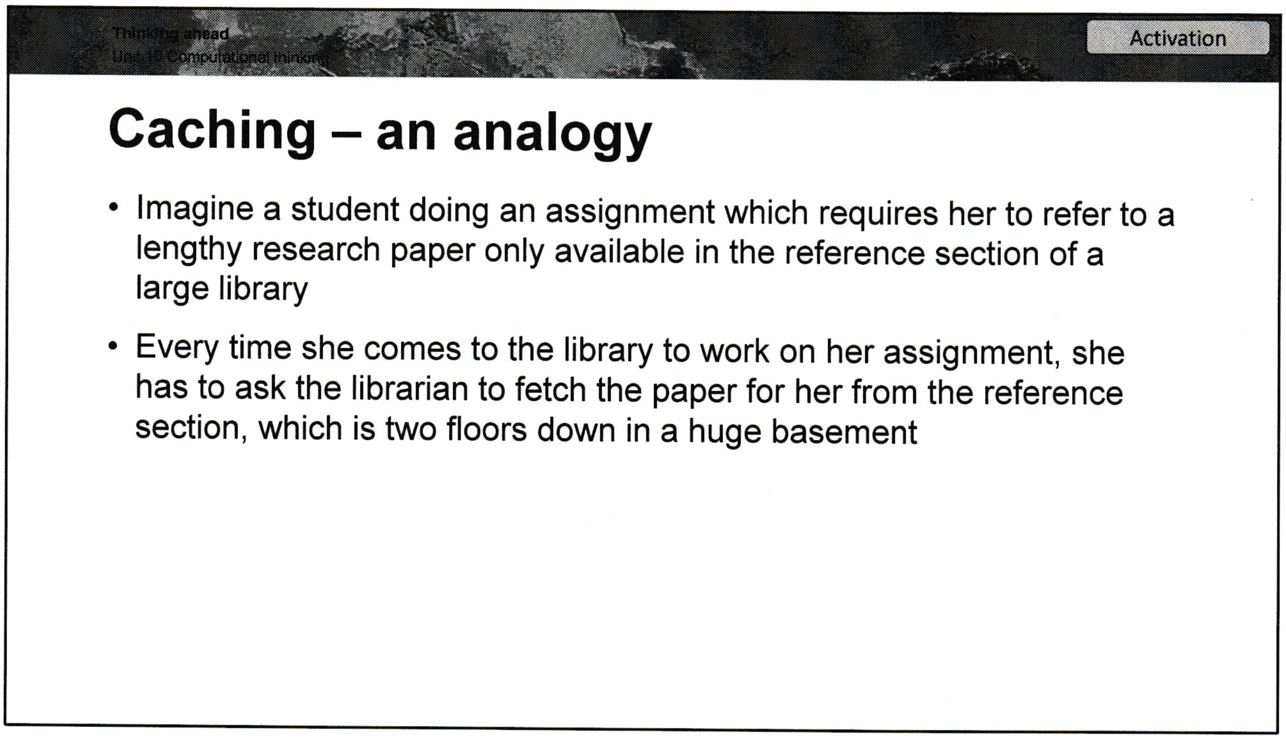
- Specifying inputs, outputs and preconditions is one aspect of thinking ahead
- Another aspect of thinking ahead, for a system designer, is trying to anticipate what the user may want to do next
- How will this impact on menu design in a software package with several levels of menu?



Activation

Caching

- Operating systems “think ahead” using caching
- Caching is the temporary storage of data and instructions
- The last few instructions of a program to be executed, the result of an earlier computation, or data used may be stored in memory so they can be very quickly retrieved
- Web caching, i.e. storing HTML pages and images recently looked at, is often used

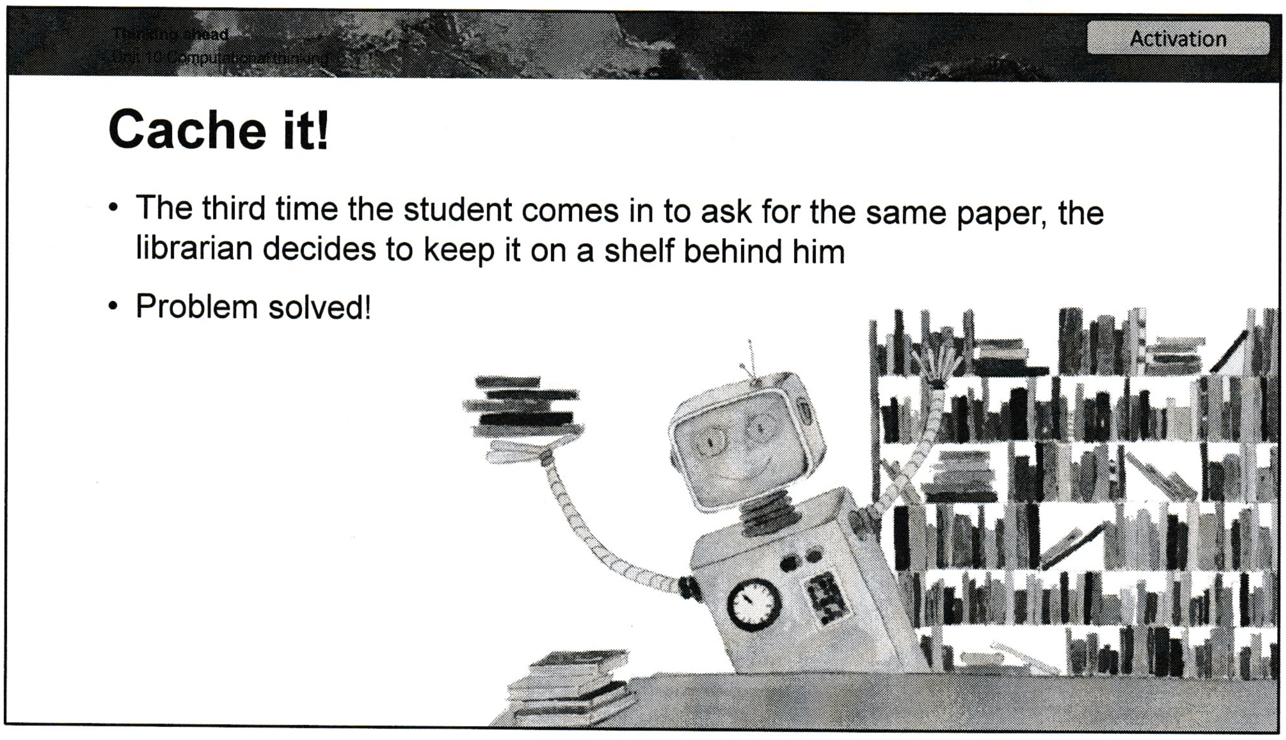


Thinking ahead
Unit 10 Computational thinking

Activation

Caching – an analogy

- Imagine a student doing an assignment which requires her to refer to a lengthy research paper only available in the reference section of a large library
- Every time she comes to the library to work on her assignment, she has to ask the librarian to fetch the paper for her from the reference section, which is two floors down in a huge basement

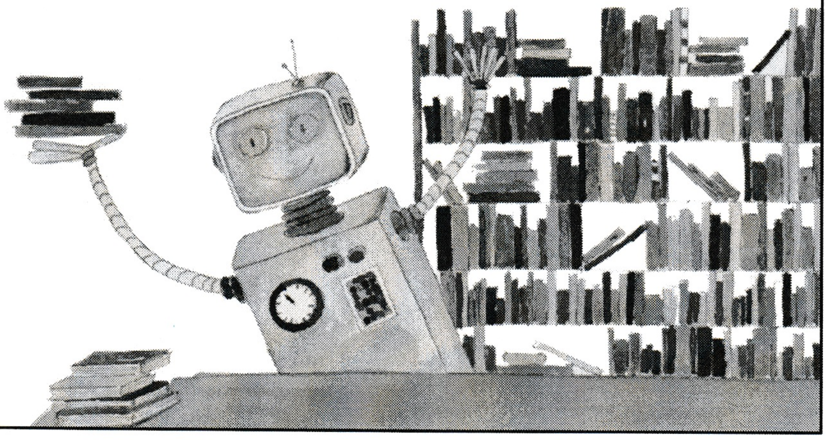


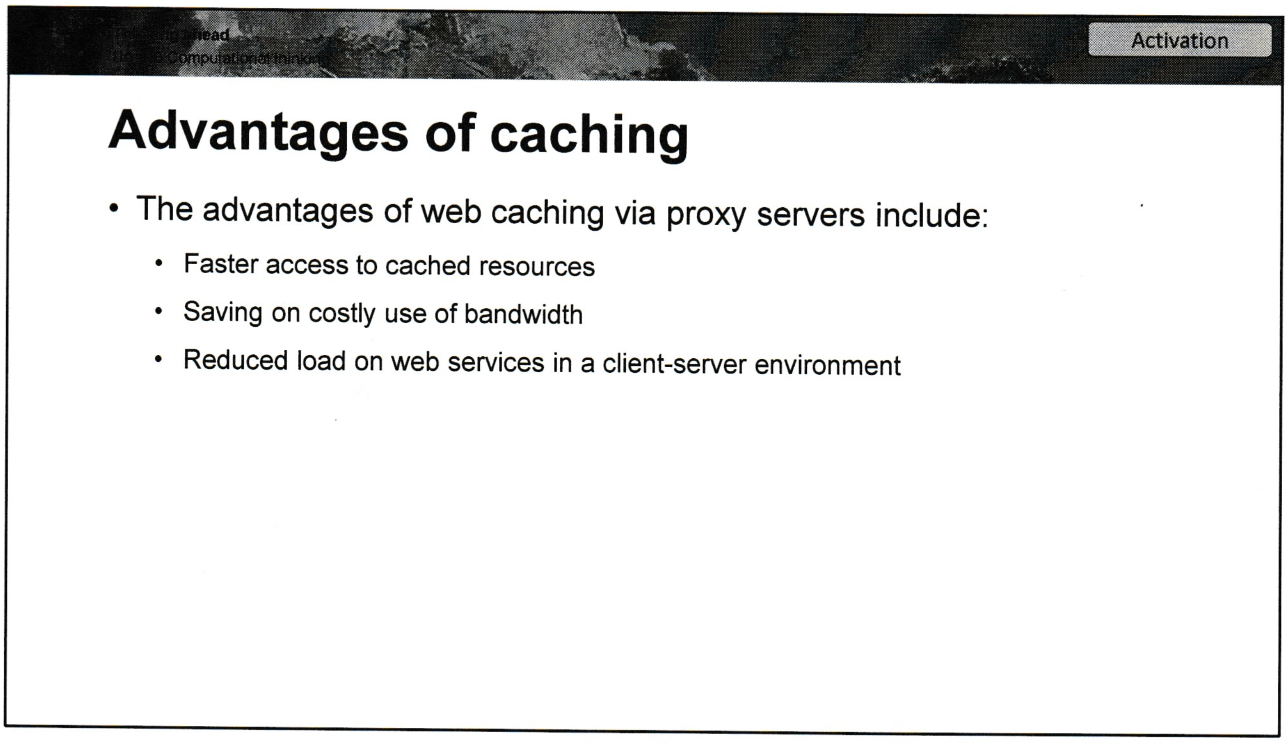
Thinking ahead
Unit 10 Computational thinking

Activation

Cache it!

- The third time the student comes in to ask for the same paper, the librarian decides to keep it on a shelf behind him
- Problem solved!



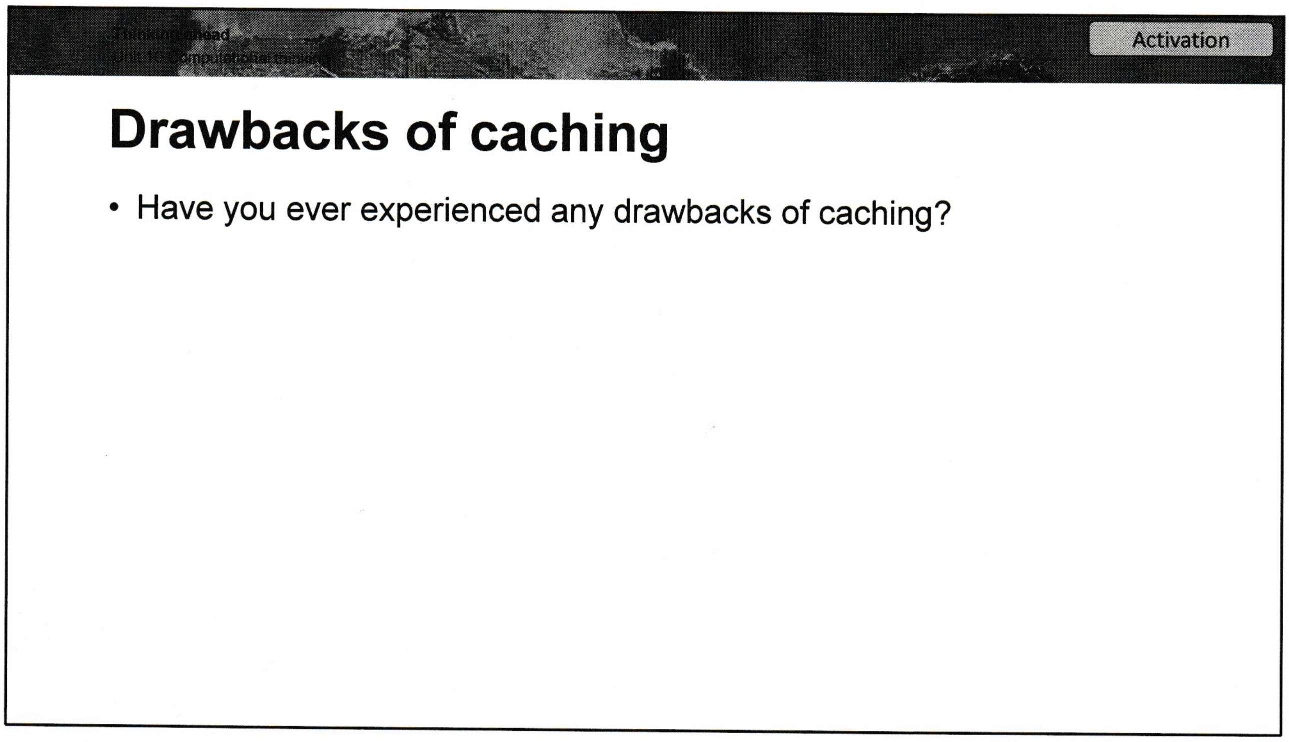


Thinking ahead
Unit 10 Computational thinking

Activation

Advantages of caching

- The advantages of web caching via proxy servers include:
  - Faster access to cached resources
  - Saving on costly use of bandwidth
  - Reduced load on web services in a client-server environment



Thinking ahead
Unit 10 Computational thinking

Activation

Drawbacks of caching

- Have you ever experienced any drawbacks of caching?

Activation

Drawbacks of caching

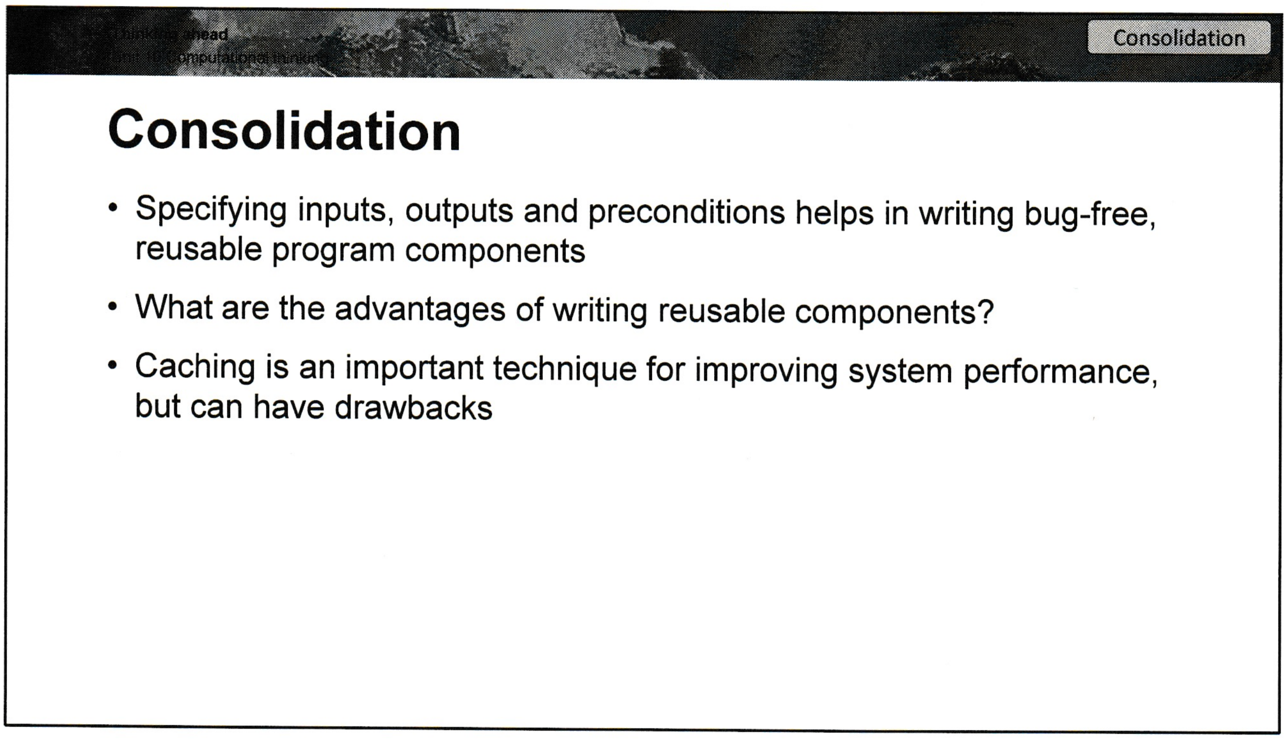
- Slower performance if the resource *isn't* found in the cache
- Being given a *stale* copy of a resource when an up-to-date copy is needed
  - For example, you access a database to get a list of available products, which is then cached
  - You make a few other queries, then a few minutes later make your original query again
  - If the query results have been cached, you may see the same available products… but in fact they may have already been sold in the meantime

Demonstration

Worksheet 2

- Do **Task 3** on Worksheet 2

Save as: S10 C48 Worksheet (your name)
Save in: Section 10

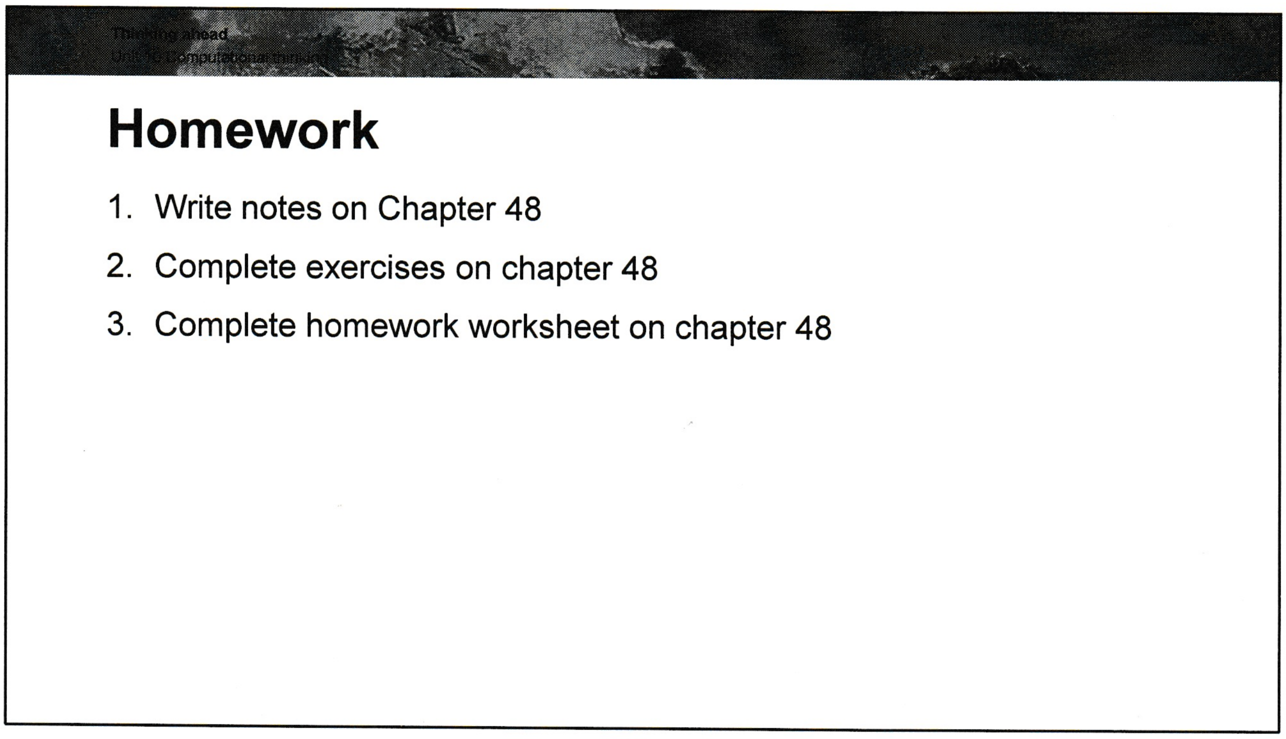


Thinking ahead
Unit 10 Computational thinking

Consolidation

Consolidation

- Specifying inputs, outputs and preconditions helps in writing bug-free, reusable program components
- What are the advantages of writing reusable components?
- Caching is an important technique for improving system performance, but can have drawbacks



Thinking ahead
Unit 10 Computational thinking

Homework

1. Write notes on Chapter 48
2. Complete exercises on chapter 48
3. Complete homework worksheet on chapter 48